Report Number: WUCS-88-11

1988-03-01

Memory Capacity of a Neural Network

Takayuki Dan Kimura

We propose to measure the memory capacity of a state machine by the numbers of discernible states, where two states are defined to be discernible if the machine manifests the identical input-output mapping in both states. According to the definition, a neuron network of n>0 inputs and one output, with an uncountable set of internal states, has the memory capacity of log2TF(n), where TF(n) is the number of different Boolean functions the network can realize with different synaptic weight and threshold values. It is shown that such a network with k>0 linear threshold units can realize at most 2k(n2+k2)... Read complete abstract on page 2.

Memory Capacity of a Neural Network

Takayuki Dan Kimura

Complete Abstract:

We propose to measure the memory capacity of a state machine by the numbers of discernible states, where two states are defined to be discernible if the machine manifests the identical input-output mapping in both states. According to the definition, a neuron network of n>0 inputs and one output, with an uncountable set of internal states, has the memory capacity of log2TF(n), where TF(n) is the number of different Boolean functions the network can realize with different synaptic weight and threshold values. It is shown that such a network with k>0 linear threshold units can realize at most 2k(n2+k2) Boolean functions and therefore the network has memory capacity of at most k(n2+k2) bits or 2k3 bits when n

MEMORY CAPACITY OF A NEURAL NETWORK

Takayuki Dan Kimura

WUCS-88-11

March 1988

Department of Computer Science
Washington University
Campus Box 1045
One Brookings Drive
Saint Louis, MO 63130-4899

Abstract

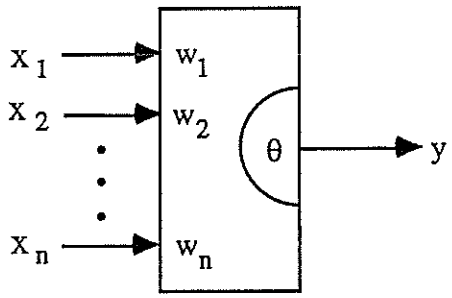
We propose to measure the memory capacity of a state machine by the number of discernible states, where two states are defined to be discernible if the machine manifests the identical input-output mapping in both states. According to this definition, a neuron network of $n>0$ inputs and one output, with an uncountable set of internal states, has the memory capacity of $\log_2 TF(n)$, where $TF(n)$ is the number of different Boolean functions the network can realize with different synaptic weight and threshold values.

It is shown that such a network with $k>0$ linear threshold units can realize at most $2^{k(n^2+k^2)}$ Boolean functions and therefore the network has memory capacity of at most $k(n^2+k^2)$ bits or $2k^3$ bits when $n<k$. This result is consistent with the previously known upper bound. We argue that a general purpose adaptive network requires an exponential number of hidden units with respect to the number of inputs n, and therefore the construction of such a network is not feasible for a large n. As a corollary, we introduce the following thesis on human perception:

> Any pattern recognition system that simulates human perception is realizable by a polynomial sized neural network.

1. Introduction

Neural networks promise a new paradigm for massively parallel computation[1]. The type of network discussed in this report is also referred to as a connectionist model[2]. As a fine-grained parallel computation model, a neural network consists of a set of adaptive processing units communicating with each other by transmission of simple numeric values. A typical processing unit, called a linear threshold unit (LTU), has the structure illustrated by Figure 1.



where $x_1, x_2, \ldots, x_n \in \{0,1\}$: the input vector

$y \in \{0,1\}$: the output

$w_1, w_2, \ldots, w_n \in R$: the connection weights

$\theta \in R$: the threshold

and the output y is a function of the input vector defined by

$$y = 1 \quad \text{if} \quad \sum_{i=0}^{n} w_i x_i \geq \theta$$

$$= 0 \quad \text{if} \quad \sum_{i=0}^{n} w_i x_i < \theta$$

Figure 1: Linear Threshold Unit (LTU)

The LTU is modeled after a biological neuron cell. The input vector represents the dendritic input and the output represents the activation state of the cell transmitted through the axon. The weight vector represents the synaptic connections. A positive weight corresponds to an excitatory synaps and a negative one to an inhibitory synaps.

---

[1] D.E. Rumelhart and J.L. McClelland, *Parallel Distributed Processing*, Vol. I and II, MIT Press, 1986.

[2] J. Feldman, "Computing with Structured Connectionist Networks," *CACM*, February 1988.

Given a particular set of weight and threshold values, the LTU realizes a n-ary Boolean function, and it is adaptive because it can represent different Boolean functions with different parameters. However, not all Boolean functions are realizable by the LTU. The LTU realizable functions are called a *linearly separable function* or a threshold *function*[3]. There exists an algorithm for finding a particular set of parameters that can realize a given threshold function.

Perceptron[4,5] is a well known neuron model based on the LRU. A perceptron may be a single LTU (a single-layered perceptron, or simply a perceptron) or a feed-forward network of cascaded LTU's (a multi-layered perceptron). A perceptron has a learning algorithm called the Delta Rule by which the synaptic weights can be modified according to a sequence of input vectors accompanied by the desired outputs. The Perceptron Convergence Theorem shows that any threshold function can be learnt by a perceptron through the Delta Rule. The utility of perceptron as a computation model is limited by the fact that it can realize only a threshold function and that the n-ary threshold functions comprise a very small fraction of all n-ary Boolean functions when n is large. Because of this limitation, a perceptron lost its significance as a general purpose computation model. A multi-layered perceptron, on the other hand, does not have such limitation, i.e., it can realize any n-ary Boolean function. However, there had been no known algorithm for training the hidden units so that the network may realize an arbitrarily chosen Boolean function.

The situation was changed by the recent discovery of the Generalized Delta Rule of the back-propagation model[6]. In spite of its local minima problem, the generalized delta rule provides a useful learning algorithm for the hidden units of the multi-layered perceptron. This discovery, along with other technical break-throughs such as Hopfield networks[7] and

---

[3] P.M. Lewis II and C.L. Coates, *Threshold Logic*, John Wiley and Sons, 1967.

[4] F. Rosenblatt, *Principles of Neurodynamics*, Sparton, 1962.

[5] M. Minsky and S. Papert, *Perceptrons*, MIT Press, 1969.

[6] D.B. Parker, *Learning-Logic*, TR-47, Center for Computational Research in Economics and Management Science, MIT, 1985.

[7] J.J. Hopfield, "Neural Networks and Physical Systems with Emergent Collective Computational Abilities," *Proc. Nat'l Acad. Sci. USA*, vol. 79, April 1982, pp. 2554-2558.

Boltzmann machines[8], renewed interests in neural network as a parallel computation model as well as an intelligent system model. In this report, we will consider a multi-layered perceptron as a representative of neural network.

An adaptive neural network can be considered not only as a switching device that implements a logical function of n variables but also as a memory device with n address lines and one data line. The content of the memory is distributed over n+1 real number parameters. Reading the content of memory at address **a** is carried out by evaluating the output of the network with **a** as the input. Writing a new value **b** at address **a** is done by adjusting the parameters by a learning algorithm in such a way that the new association between **a** and the desired output **b** is realized without disturbing the other existing associations. If the network is capable of realizing any n-ary Boolean function and there exists a learning algorithm to find corresponding parameters, then the memory capacity of such a network is 2^n bits.

In order for neural network to be a viable alternative to the existing paradigms for parallel computation, we have to resolve some technical issues. One of them is the feasibility of constructing an universal neural network, i.e., a general purpose adaptive network that can learn to represent any n-ary Boolean function, or equivalently to have the memory capacity of 2^n bits. A related issue is how to find the minimum number of hidden units necessary to realize a given Boolean function. At present there is no known algorithm to solve this problem, and only experimental results are known.

In this report we will address the first issue of an universal network feasibility. In particular, we are interested in measuring the memory capacity of a neural network consisting of $k > 0$ processing units. In the next section we will make the concept of memory capacity more precise. Section 3 shows that the memory capacity of $n > 0$ input neural network with $k > 0$ processing units is at most $k(n^2+k^2)$ bits or $2k^3$ bits when $n < k$. In Section 4, we will discuss the significance of the results for the future research in the area of neural network.

---

[8] D.H. Ackley, G.E. Hinton, and T.J. Sejnowski, "A Learning Algorithm for Boltzmann Machines," *Cognitive Science*, vol. 9, 1985, pp. 147-169.

2. LTU as a Memory Device

Consider an arbitrary automaton M with $n > 0$ binary inputs, one binary output, and the set S of possible internal states. We are interested in the question: what is the memory capacity of M, i.e., how much information M can store by changing its internal state. We define the memory capacity of M by the the number of internal states that are discernible from the environment. Note that for each state $s \in S$, there exists an n-ary Boolean function f_s that maps each input vector to a binary value. We define two internal states s_1 and s_2 of S to be *equivalent* ($s_1 \equiv s_2$) if $f_{s_1} = f_{s_2}$, and *discernible* if they are not equivalent. Formally, the memory capacity of M, measured in bits, is defined as:

$$\log_2 \#(S/\equiv),$$

where $\#(S/\equiv)$ is the cardinality of the partition on S induced by the equivalence relation $\equiv$. For example, if all the internal states of M are discernible from the environment, then the memory capacity of M is exactly $\log_2(\#S)$ bits, where #S denotes the cardinality of S.

The LTU of Figure 1 is an automaton with the uncountable set R^{n+1} of internal states, where R is the set of real numbers. Each state $s \in R^{n+1}$ corresponds to a particular set of weight and threshold values, and $s_1 \equiv s_2$ if and only if s_1 and s_2 realizes the same Boolean function. Therefore, $\#(S/\equiv)$ is equi-numerous to the number of n-ary Boolean functions realizable by the LTU, i.e., the number of n-ary threshold functions.

Let TF(n) be the number of n-ary threshold functions. Finding the closed formula for TF(n) has been a long-standing open problem. However, the following upper-bound has been known[9,10]: For $n \geq 2$,

$$TF(n) < 2\sum_{k=0}^{n} \binom{2^n - 1}{k} < \frac{2^{n^2+1}}{n!}.$$

Since $\frac{2}{n!} < 1$, for any $n \geq 3$, we will use the following upper-bound for TF(n):

$$TF(n) < 2^{n^2} \quad \text{where } n \geq 3.$$

---

9 S.H. Cameron, "An Estimate of the Complexity Requisite in a Universal Decision Network," *Bionics Symposium*, WADD Report 60-600, December 1960.

10 R.O. Winder, *Threshold Logic*, Ph.D. Thesis, Mathematics Department, Princeton University, 1962.

Therefore, the memory capacity of a single LTU is at most n^2 bits, i.e.,

$$\log_2 TF(n) < n^2.$$

The Table 1 below compares the number of n-ary threshold functions with the total number of n-ary Boolean functions, BF(n), and the various upper-bounds[11].

n	$BF(n) = 2^{2^n}$	TF(n)	$\frac{2^{n^2+1}}{n!}$	2^{n^2}
1	4	4	4	2
2	16	14	16	16
3	256	104	171	512
4	65,536	1,882	5,461	65,536
5	4,294,967,296	94,572	559,241	33,554,432

Table 1: The number of Threshold Functions

Note that the ratio TF(n)/BF(n) asymptotically approaches zero, and that for a biological neuron cell, n ranges between 7,000 and 50,000.

3. Memory Capacity of A Neural Network

In this section we will compute an upper bound for the memory capacity of an $n > 0$ inputs, one output, layered neural network of $k > 0$ LTU's.

Winder[12] obtained the following upper bound for the number of functions realizable by a general network of $k > 0$ threshold units with no feedback and $n > 0$ inputs:

$$\frac{2^{n(nk+\frac{k(k-1)}{2})+k}}{(n!)^k}.$$

We will derive a simpler form of upper bound for a multi-layered network. Our result is consistent with the above in the sense that a network of k units can be shown by the both results to have the memory capacity of $O(k^3)$ bits.

---

[11] P.M. Lewis II and C.L. Coates, *Threshold Logic*, John Wiley and Sons, 1967.

[12] R. O. Winder, "Bounds on Threshold Gate Realizability," *IRE Transasction on Electronic Computers*, EC-12,(5), 1963, pp. 561-564.

Consider the m > 0 layered neural network of Figure 2.

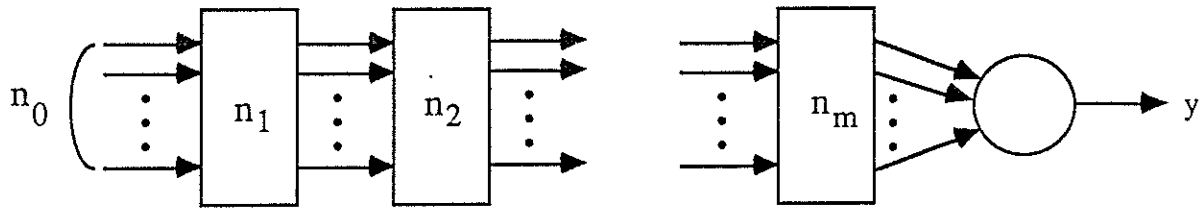


Figure 2: A Layered Neural Network

The first layer consists of n_1 units each of which has n inputs, and it generates n_1 values as outputs which are in turn fed into the second layer. The second layer consists of n_2 units each of which has n_1 inputs, and it generates n_2 values fed into the next layer, and so forth. The m-th layer consists of n_m units generating n_m values for the output unit. Thus, the total number of processing units in the network is:

$$k = n_1 + n_2 + \ldots + n_m + 1, \qquad \text{where } n_i > 0 \text{ for } 1 \le i \le m.$$

Let α be the number of Boolean functions realizable by the above network. Then the memory capacity of the network is $\log_2\alpha$. Using the notation given in the previous section, we can express the number of functions realizable by the n_i units of the i-th layer as

$(\, TF(n_{i-1}) \,)^{n_i}$ where $n_0 = n$, and $1 \le i \le m$,

and $TF(n_m)$ for the output unit.

Therefore, the total number of functions realizable by the entire network is:

$$\alpha = \Big(\prod_{i=1}^{m} (\, TF(n_{i-1}) \,)^{n_i} \Big) \times TF(n_m).$$

Using the upper-bound obtained in Section 2, i.e., $TF(n) < 2^{n^2}$ $(n \ge 3)$,

$$\alpha < \Big(\prod_{i=1}^{m} \big(2^{n_{i-1}^2 \times n_i} \big) \Big) \times 2^{n_m^2}.$$

It follows that

$$\log_2 \alpha < \Big(\sum_{i=1}^{m} (n_{i-1}^2 \times n_i) \Big) + n_m^2$$

$$< k \times n_0^2 + k \times \sum_{i=1}^{m} n_i^2$$

$$< k\,(n_0^2 + k^2) = k\,(n^2 + k^2).$$

Note that for $1 \le i \le m$, $n_i < k$ and $n_i^2 < k^2$ by the definition of k.

Therefore, the number of Boolean functions realizable by the network is no more than $2^{k(n^2+k^2)}$ and the network's memory capacity is at most $k(n^2+k^2)$ bits. When $n < k$, the memory capacity is at most $2k^3$ bits.

4. Conclusions

One corollary of the polynomial upper bound for the memory capacity is that a construction of a universal neural network with $n > 0$ inputs is not feasible for a large n, because it requires an exponential number of hidden units with respect to n. By a universal neural network we mean a network that can learn any n-ary Boolean function. If a single-layered perceptron lost its significance because of its limitation on function realizability, a multi-layered perceptron suffers the same problem as a general computation model.

By the same token, a neural network with a polynomial number of hidden units may not be able to produce a solution mapping, because such a network can realize only a very small portion of the possible mappings. This also means that when a training sequence fails to produce a solution, it is impossible to decide whether the failure is due to insufficient training or due to the limitation of the network's function realizability.

On the other hand, it is reasonable to assume that most applications, such as pattern recognition problems and image processing applications, require implementation of a logical function that is realizable by a polynomial sized neural network. For example, the parity function, which is known to be a difficult function to realize by a LTU network, may not be a significant function for simulating human perception. One reason is that human perception cannot evaluate the parity function easily and efficiently, as illustrated by the two pictures of Figure 3. It is almost impossible to discover, by perception only without logical reasoning,

that the two pictures have different parity values. What human perception cannot achieve may not be achievable by its artificial simulator.

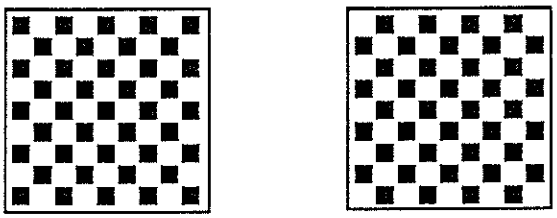


Figure 3: Patterns Discernible by the Parity Function

Accordingly, we introduce the following thesis that enables us to continue future research on neural network applications:

> Any pattern recognition system that simulates human perception is realizable by a polynomial sized neural network.

From the viewpoint of parallel computation modelling, a neural network provides a paradigm of non-symbolic computation that is diametrically different from traditional symbolic paradigms. In this new paradigm information is carried by continuous patterns instead of discrete symbols, and the fundamental operation is superposition instead of concatenation. The functional upper bound, however, indicates that a neural network cannot be a solution to all computational problems. It also indicates that a large number of processing units are needed for a complex computation and the large scale concurrency is a necessity rather than a choice.